Modeling Public Business Processes with Interaction Protocols in Business-to-Business Relationships

Pablo David Villarreal[1], Enrique Salomone[1,2], Omar Chiotti[1,2]

[1]GIDSATD Research Group
Universidad Tecnológica Nacional - Facultad Regional Santa Fe
Lavaisse 610 - (3000) Santa Fe - Argentina

[2]INGAR-CONICET
Avellaneda 3657 – (3000) Santa Fe – Argentina

pvillarr@frsf.utn.edu.ar;{salomone,chiotti}@ceride.gov.ar

Abstract

Today, one of the main challenges of the Business-to-Business (B2B) E-Commerce solutions is to improve the business processes that organizations carry out with their suppliers and customers. B2B relationships require the management of public business processes, which span the organizational boundaries and are jointly managed by the organizations. Although Workflow and Web Service Composition approaches have been proposed to model and manage public processes, they present shortcomings to achieve the requirements of autonomy, decentralization, P2P interaction and negotiation. A flexible approach that can achieve these requirements is the use of *Interaction Protocols* to model and manage public processes. Interaction Protocols allows to model public processes with a high abstraction level, which can be understood by the enterprises, people and systems.

In this work we describe the modeling of public processes using interaction protocols. We discuss different aspects for representing interaction protocols in the context of B2B relationships. In addition, we describe several examples of public processes modeled with interaction protocols. These examples of public processes are derived from collaborative models for the supply chain management. Benefits of the interaction protocols for represent public processes in B2B relationships are also identified.

Keywords: Business-to-Business, Business Process, Workflow, Interaction Protocol.

1. Introduction

Today, one of the main challenges of the Business-to-Business (B2B) E-Commerce solutions is to improve the business processes that organizations carry out with their suppliers and customers. B2B relationships require the management of two types of business processes: private processes and public processes. *Private processes* are own processes of the enterprises and they are managed by each enterprise in an autonomous way. Private processes are supported within the enterprises using traditional Workflow Management Systems (WfMSs) [19], ERP systems or proprietary systems. Instead, *public processes* span organizational boundaries. They belong to the enterprises involved in a B2B relationship and are agreed and jointly managed by the partners. Activities of public processes are supported by private processes. A clear separation between private and public processes enables to the organizations to abstract the management of their internal processes from the management of processes across enterprises.

Collaborative business models have been proposed in several application domains, like supply chain management. They include public business processes and are supported by B2B relationships. Some collaborative models [16] require organizations to establish an independent B2B relationship with each partner, allowing organizations collaborate in a partner-to-partner way. From an information systems point of view, these collaborative models impose several challenges to support the management of public business processes in B2B relationships:

- Autonomy, which implies that the enterprises should be able to behave as autonomous entities, hiding their internal decisions, activities and processes. Information systems that manage the B2B relationships in each enterprise have to be independent.
- Decentralized management of the business processes jointly managed by the enterprises.
- Peer-to-Peer interactions among the systems that manage B2B relationships. This means that systems have to interact in a direct way without the mediation of a third party system.
- Negotiation has to be included in the public process management.

Workflow and Web Service Composition approaches have been proposed to model and manage public processes [12]. However, they present shortcomings to achieve the above requirements [18]. Therefore, with the purpose of using a suitable approach to fulfills the requirements above established, in [18] we have proposed the use of *Interaction Protocols* to model and manage public processes in B2B relationships. Interaction Protocols have been used in the area of multi-agent systems to represent interactions among agents [1]. In the context of B2B relationships, interaction protocols represent interactions among enterprises within a process that they jointly carry out to achieve a common goal. An interaction protocol describes a high-level communication pattern through an admissible sequence of messages between enterprises having different roles. The main objective of interaction protocols is to abstract public processes from the services to be invoked within each enterprise for executing the internal activities supporting public processes.

In this work we identified different aspects required to model public processes using interaction protocols in B2B relationships. Moreover, we describe the modeling of different public processes, from collaborative e-business models, using interaction protocols. Section 2 discusses different aspects required to represent interaction protocols in B2B relationships. Section 3 describes the definition of interaction protocols that model public processes of collaborative models. Section 4 evaluates the benefits of the interaction protocols to model public processes. Section 5 presents conclusions and future works.

2. Representing Interaction Protocols for B2B Relationships

In our research, interaction protocols represent interactions among enterprises in B2B relationships.

These interactions model the public processes that enterprises have to jointly manage, providing a high abstraction level. A message of an interaction protocol does not represent a message on the network. Messages are implemented using a low-level communication protocol. Therefore, interaction protocols are different from the communication protocols. Interaction protocols define messages with their content but not their structure, and they do not also provide the way in which the messages are packaged and transported on the network.

The main elements of an interaction protocol are: roles, messages, conditions, control flows, logical connectors and deadlines. A *role* represents the responsibility, in terms of a messages sequence that an enterprise performs in a B2B relationship. *Messages* express interactions and contain a semantic defining their type. A message can represent business information, decision, proposal, acceptation, rejection or acknowledgment. A message can be asynchronous or synchronous. *Conditions* can be defined on messages to represent when a message can be sent. An interaction protocol has two *control flows*. One represents the control flow of the messages, which defines the parallel or alternative messages of each interaction protocol step. The second represents the internal execution flow of a role that describes the different reactions of the role to the incoming messages. Basic *Logical connectors*, as AND, XOR, and OR, are used to define control flows. *Deadlines* can be defined on messages representing the time that a role has to send a message.

Following, we discuss several aspects required to model interaction protocols in B2B relationships.

2.1. Graphical Languages for Modeling Interaction Protocols

To enhance the business process design activity, a graphical modeling language is required to define interaction protocols. Some modeling techniques from software engineering have been applied to business processes, especially UML [13]. Adaptations and extensions on UML have been proposed to describe the structural and behavioral aspects of a business [7]. Behavioral aspects can be captured with UML through sequence diagrams, collaborations diagrams, activity diagrams and states diagrams. Activity diagrams and states diagrams have been used to model business processes. However, these UML diagrams are not suitable to model interaction protocols because they do not provide all the notations required to represent the elements of the interaction protocols.

Other graphical languages or notations that can be candidate to specify interaction protocols are: AUML [1], Petri Nets and statecharts. Their advantages and disadvantages have been studied in the context of agent interactions in agent-oriented software engineering [14]. Although in this paper we do not focus on determining the most appropriate language to model interaction protocols, we have selected AUML due to it has several benefits to define business processes in B2B relationships:

- AUML provides enough graphical expressiveness, so interactions protocols are easy to read.
- Interaction protocols elements are shown explicitly (e.g. messages exchange with arrows).
- AUML provide an explicitly representation of the messages over the time through lifelines.
- AUML is based on UML, so it is familiar to UML users that use it for software engineering or for modeling business processes. In this way, AUML could facilitate communication between business processes modelers and software modelers.

In AUML, interaction protocols are modeled through protocol diagrams, which are an extension of the sequence diagrams of UML 1.x [13]. Roles are represented by a rectangular box indicating the enterprise that performs the role. Role lifeline defines the time period during which the enterprise participates in the protocol. Lifelines are represented by dotted vertical lines. The lifeline may split up into two or more lifelines to show AND and OR parallelism and decisions, corresponding to branches on the incoming message flow. Logical connectors are also used to define parallelism and decisions in the message flow. A thread of interaction in AUML, which is represented by a bar on the lifeline, shows the period during which the enterprise role is performing some private activities or processes to react to an incoming message or to send a message. An asynchronous message is drawn as ⟶. A synchronous message is drawn as ⟶. Deadlines are represented by comments on the arrows. Conditions can be added on messages using brackets.

2.2. Semantic of the Message Types

Messages of interaction protocols should be understood in the same way by the enterprises involved in a B2B relationship. This can be achieved through the use of a common semantic for the specification of messages. This semantic should be clear and should define the complete set of message types that can be used in B2B interactions. The message types should represent different intentions that an enterprise wishes to express in the processes. Intentions are important to describe negotiations. Besides, the semantic of the messages should be based on formalisms and should be separated of the message content definition. Message content is defined by business document types. The semantic of the message content is different and independent from the semantic of the message type. The semantic of the message content refers to the common knowledge model and vocabulary necessary to understand in a same way the information exchanged by the partners [16].

Several B2B standards can be used to define messages of the interaction protocols. Some B2B standards provide a set of message types together with their content, like RosettaNet (www.rosettanet.org). Other B2B standards provide a set of verbs that can be used to define messages, like OAGIS (www.openapplications.org). Other alternative is to define and agree between the partners the semantic of the message types, which can be defined through a B2B standard, like ebXML (www.ebxml.org) or OAGIS. These B2B standards can be used to define particular messages and business documents.

However, the above alternatives are not suitable because the interaction protocols definitions cannot be reused. To do so, we can use a rich set of message types like the common and possible messages to be used in the interactions. These message types should be extensible and should provide the basic message types suitable to express any type of business process, including negotiation processes. The objective of this set of message types is abstracting the protocol definition from the specific B2B standards. Today, there is no consortium that provide a B2B standard with message types that satisfy our semantic requirements. This is mainly due to most of the B2B standards provide generic specifications with the purpose of defining any type of business processes.

Therefore, as set of message types we propose to use the communicative (speech) acts or performatives defined by FIPA ACL (Agent Communicative Language) [8], which is a standard language used to express interactions among software agents. This set of communicative acts fulfills the requirements we pursue with respect to the semantic of the interaction protocol messages. Communicative acts allow enterprises define their intentions in the public processes and they can be defined in some B2B standard like ebXML or OAGIS, in which the messages can be defined using communicative acts.

2.3. Templates of Interaction Protocols

To enhance the reuse and the design activity in the modeling of interaction protocols, an important practice is the use of protocol templates. They can be customized to define an interaction protocol, assigning values to the set of parameters of the protocol template. The parameters can refer to the message content, deadlines and variables used in conditions of the messages.

Examples of protocol templates are those proposed by FIPA to communicate agents, like the Contract Net, Request and Query protocols [9]. These templates are independent of the problem domain. However, enterprises can define their own templates based on their problem domain. In addition, enterprises can define a protocol template for reusing of interaction protocols.

2.4. Implementing Interaction Protocols with B2B Standards

B2B standards are used to exchange messages among enterprises in B2B relationships. Therefore, B2B standards can be used to implement and exchange messages defined in the interaction protocols. In this way, enterprises can implement independent and different systems to jointly manage the same interaction protocols without the use of a proprietary low-level communication protocol. There are a large number of B2B standards, which can consist of the specification of several elements [3]. One element is the machine-processable textual definition language to define public processes. Another element is the exchange sequence, which defines the possible transactions required by each message and the constraints on the message, like time-outs and performance. Most of the B2B standards also specify the syntactic structure of the business documents involved in the message content. Moreover, some B2B standards provide the form in which the messages are packaged and transported on the networks using a particular communication protocol, like HTTP, SMTP and so on. Another element that B2B standard can provide is the security mechanism for the messages.

These elements are required to implement interaction protocols. The B2B standards that provide most of these elements are ebXML (www.ebxml.org), RosettaNet (www.rosettanet.org) and OAGIS (www.openapplications.org).

3. Modeling Public Business Processes with Interaction Protocols

The main difference of interaction protocols with other approaches to model public processes is that interaction protocols do not define activities or services. Modeling processes with interaction protocols we mainly focus on the messages that have to be sent and received by the enterprise roles in each step of the business process.

Based on the discussed in the above section, following we describe how it is possible to model public business processes with interaction protocols. We use as examples, public processes derived from collaborative models proposed on the application domain of the supply chain management.

3.1. Example of a CPFR Process Implemented with RosettaNet

The Collaborative Planning, Forecasting and Replenishment (CRPF) model (www.cpfr.org) is an initiative of the retail industry. CPFR consists of four main processes: *Business Planning*, *Strategic Forecast*, *Order Forecast* and *Replenishment*. It also contains *Exception Management* processes.

RosettaNet is a B2B standard defined for the electronic components supply chain. A guideline has been published to describe the use of RosettaNet Partner Interfaces Processes (PIPs) to implement CPFR processes [15]. We describe how the *Collaborative Order Forecast Process* defined in the guideline could be defined with interaction protocols. In [15], this process is defined in an informal workflow showing the activities that enterprises have to carry out (see Figure 1).

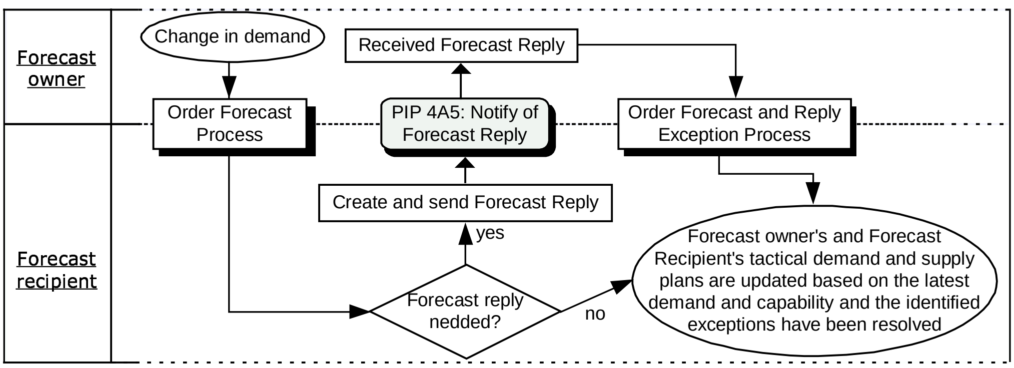

Figure 1. Workflow of the Collaborative Order Forecast Process using RosettaNet.

The first activity is the *Order Forecast Subprocess* that is initiated by the forecast owner when detects changes in demand. This subprocess represents three types of *Order Forecast Process* that can be executed according to the replenishment model used. Their details are not showed here. We only need to know the exchange messages by the partners. RosettaNet PIPs 4A2, 4A3 and 4A4 are the exchanged messages for each *Order Forecast Process* type. In addition, this subprocess contains an activity that represents another subprocess to manage exceptions that can occur on the order forecast. For more details about the *Order Forecast* and the *Exception Processes* see [15].

As we mentioned above, to define the interaction protocol we consider the messages to be exchanged between the enterprise roles. The messages are derived from the RosettaNet PIPs involved in the process. RosettaNet PIPs define interaction points that messages represent in interaction protocols and they are also used to implement messages of the interaction protocols.

The interaction protocol that defines the *Collaborative Order Forecast Process* is shown in the Figure 2. The protocol starts with the forecast owner role, which can send three types of *Order Forecast* according to the replenishment model agreed. Then, partners can perform the *Exception Management* nested protocol if an exception condition occurs. Exceptions and their conditions are established in the collaborative agreement. This nested protocol is defined based on the *Exception Management* protocol template. It was defined to manage exceptions in different forecast processes. Therefore, this protocol template can be reused in different public processes. At the bottom of the nested protocol there is a condition indicating if it is evaluated as true, the roles will perform the nested protocol. The role that monitoring and informs the exceptions is defined previously in the agreement. In our case, this is performed by the role *forecast owner* who is the initiator role.

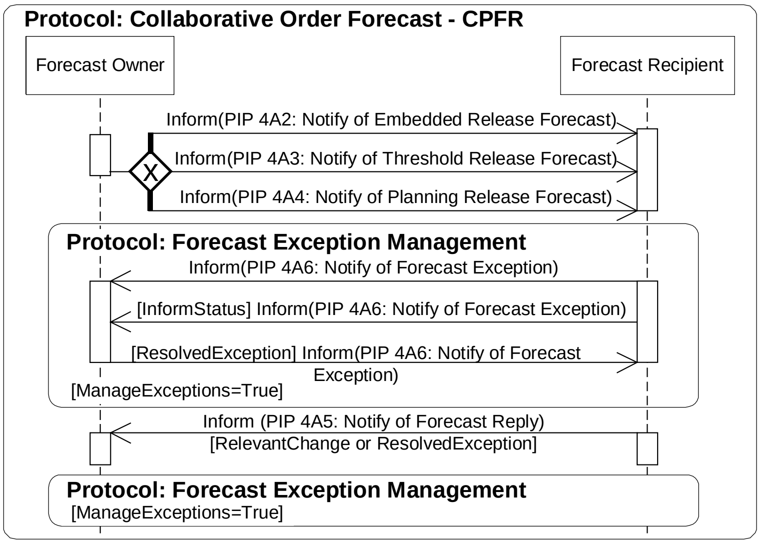

Figure 2. Interaction Protocol of the Collaborative Order Forecast Process using RosettaNet

In the next step, the *Forecast Recipient* can send a *Forecast Reply* if it determines that relevant changes have occurred or if an exception has been resolved. This is indicated through a condition on the message. Then, partners can again perform the *Exception Management* protocol if an exception occurs on the *Forecast Reply*.

For reusing this protocol, we specify the messages using FIPA ACL. In this way, we can implement this protocol with any B2B standards. Due to the original process uses RosettaNet PIPs, we define this protocol encapsulating RosettaNet PIPs with FIPA ACL performatives in the messages.

3.2. Example of a SMI Process from the EIDX Association

In a Supplier Managed Inventory (SMI) model, also known as Vendor Managed Inventory (VMI), the supplier determines when to ship materials and how many units to ship for replenishing the inventory of the customer. Calculation of the replenishment may be based on a forecast or only on consumption data. As example, we describe a public business process of the forecast-based SMI model defined by EIDX (Electronics Industry Data Exchange Association). EIDX develops business process models, which are independent of the B2B standards.

Figure 3 shows the replenishment process of the forecast-based SMI model. This process has been defined by EIDX following a workflow approach with an UML activity diagram [6], where activities and exchanged documents by each partner are specified. The process has two subprocess: *Blanket Purchase Order and Release Against Blanket PO*. Their details can be found in [6].

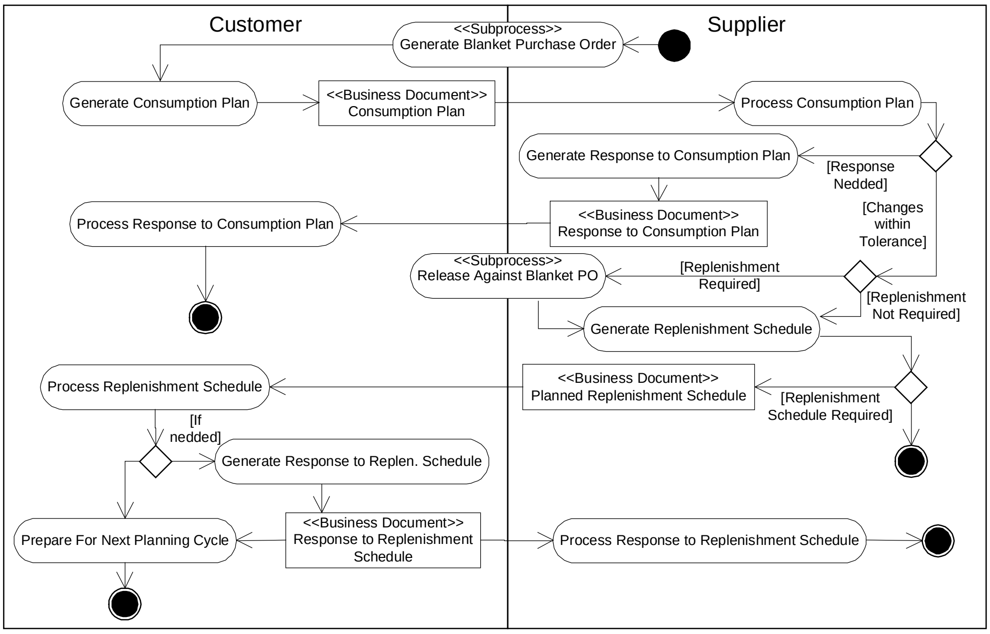

Figure 3. Workflow of the Replenishment Process - Forecast-Based SMI

This process can be specified through an interaction protocol in the following way. The interaction protocol is shown in the figure 4. The protocol starts with the nested protocol *Blanket Purchase Order*, which represents the subprocess with the same name. This nested protocol starts with the customer informing a *Blanket Purchase Order (BPO)*, which is a long-term commitment to a supplier for material against which multiple short-term releases will be generated to satisfy requirements. The *BPO* defines specific terms, conditions, and pricing terms. Then the supplier has to response with an acknowledgment to the customer, representing that the *BPO* has been received and successfully processed by the private systems. The indication that a message requires an acknowledgement is represented using a circle on the beginning of the message arrow. In this way, it is not necessary define another message representing the acknowledgment of an above message.

If the supplier determines that changes on the *BPO* are required, he requests a change. Then the customer evaluates the *BPO change* requested and can decide three alternatives. One alternative is to refuse the change requested if it determines that changes on the *BPO* are not necessary or if it does not agree with the change requested by the supplier. Another alternative is to agree with the supplier on the change requested. An a third alternative is to propose a counterproposal. In this case, the supplier evaluates the new *BPO* and can again request a change.

If the supplier does not request any change, the customer can also initiate changes on the *BPO* and send it to the supplier though a *propose* message. To express that changes to the *BPO* can be done by the customer or by a request from the supplier, we define an XOR logical connector on the customer lifeline. It represents that one of the two lifelines can be instantiated.

This nested protocol can finish in different ways. One is when the customer informs the *BPO*, it does not do any changes and the supplier does not request changes. Another one is when the customer refuses or agrees with the supplier on the changes requested. A last way is when the customer proposes a change and the supplier does not respond. The second way have an explicit and understood end, but in the other ways it is not easy to understand the end of the process. So, we add deadlines to this process. They indicate that some alternative have to be done before two days, after the customer informed the BPO. If any alternative does not occur before the deadline, the nested protocol ends and the process continue the execution of the next messages.

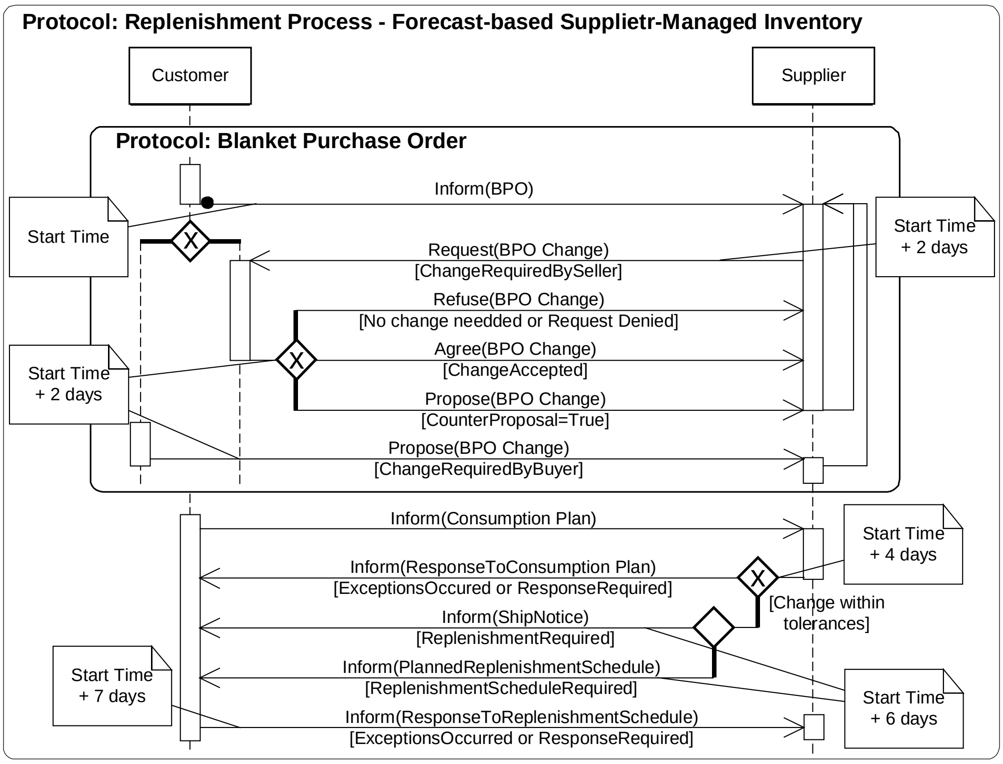

Figure 4. Interaction Protocol: Replenishment Process – Forecast-Based SMI

Once the nested protocol has been performed, the customer informs the *Consumption Plan*, which contains a planned consumption (sales and/or usage and gross requirements), available inventory levels, inventory receipts, and minimum and maximum inventory level targets. Based on the acceptable tolerances for the consumption plan defined in the agreement, if the supplier detects exceptions on the consumption plan, it will informs a *Response to Consumption Plan* to the customer. Else, the supplier can send two messages.

On the one hand, the supplier can inform a *Ship Notice* if replenishment is required during the current planning cycle. In the original workflow, this is represented with the *Release Against Blanket PO* subprocess. However, it was not necessary to define this subprocess as a nested protocol because an only message can be exchanged. On the other hand, based on the data of the customer's inventory and consumption plan, the supplier creates and informs a *Replenishment Schedule*. Then, if the customer receives this message, it can inform a *Response to Replenishment Schedule* when exceptions have been detected on the *Replenishment Schedule*.

Deadlines are added for some messages to provide a well-defined protocol. The deadlines are useful when we have messages that have a condition. If these messages do not occurs before the deadlines, the execution continues with the next messages and should not come back.

3.3. Example of a Negotiation Process from the P2P Collaborative Model

The Partner-to-Partner Collaborative Model [17] was proposed to carry out business processes among manufacturing enterprises. The model consists of public business processes that partners jointly carry out in a decentralized way.

We describe an interaction protocol defining the public process *Collaborative Production and Capacity Planning* from the Collaborative Partner-to-Partner Model, which is shown in the Figure 5. This interaction protocol has two roles, supplier and customer, that are performed by enterprise A and enterprise B respectively. The objective of this public process is that enterprises A and B agree on a *Mid-range Supply Plan* of a certain product.

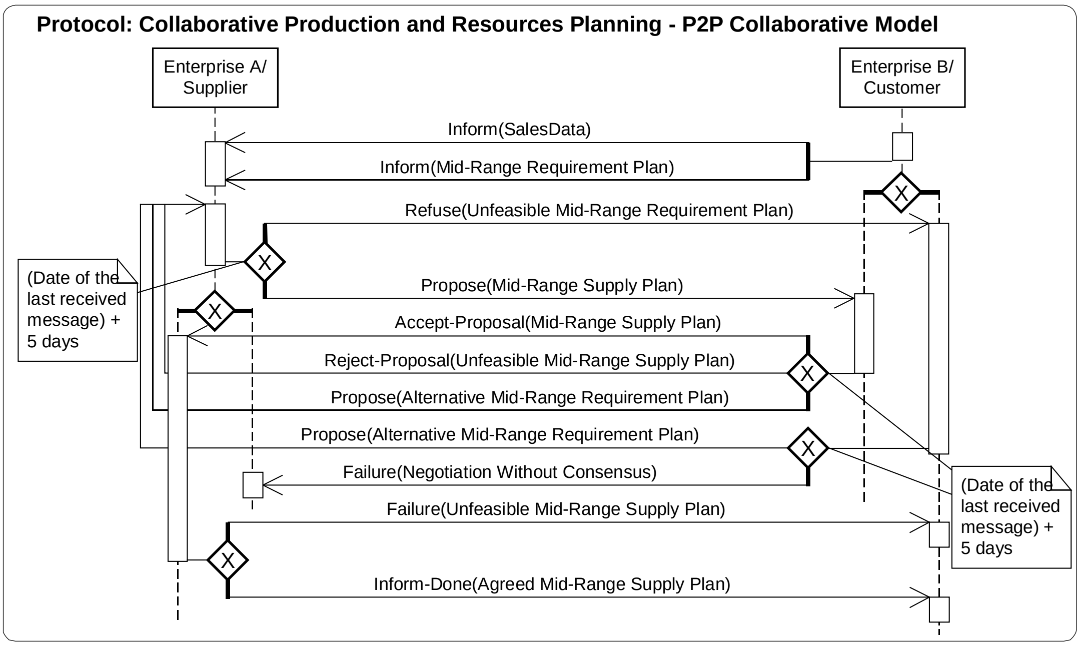

Figure 5. Interaction Protocol: Collaborative Production and Resources Planning Process

The process starts with the customer sending two parallel messages with business information: *inform(SalesData)* and *inform(Mid-Range Requirement Plan)*. Once the supplier receives these messages, its private processes are invoked for reviewing the internal *Aggregated Production Plan* and then this plan is compared with the customer plan. After comparing the plans, the supplier decides whether to propose a *Mid-Range Supply Plan* or refuse the customer plan because it cannot be satisfied. Only one of the two alternative messages can occur. This decision has to be done before the time indicated by a deadline.

The customer has two interaction threads that represent the incoming messages. If the supplier proposes a plan, the customer executes its private processes and can decide three alternatives: accept the supplier plan, reject the supplier plan, or make a counterproposal offering an alternative plan. The last two alternative messages represent a repetition of messages. This means that the supplier again executes its private processes to determine whether to propose a plan or to reject the customer plan. In the other case, when the customer receives a message *refuse (Mid-range Requirement Plan),* the customer can propose an *Alternative Mid-Range Requirement Plan* or can inform a *failure* in the negotiation because consensus has not been achieved. In both cases, the customer has to respond before a deadline.

Finally, if the customer accepts the supplier plan, the supplier can inform if the status of the supply plan passes from proposed to agreed. This occurs if the supplier still considers this plan as feasible. Otherwise, the supplier informs a failure indicating that the supply plan cannot be satisfied.

4. Evaluation of the Interaction Protocols to Model Public Processes

Workflow Management Systems (WfMSs) [19] have been used successfully by organizations to manage their private processes. However, traditional WfMSs are not appropriate to manage public business processes [5]. For this purpose, a number of approaches have been proposed based on adaptations or extensions to the traditional workflow concepts. These approaches can be classified in [18]: Split and Deploy, Composition, Subcontracting, and Public and Private. The first three approaches do not fulfill the requirements of collaborative models because they perform a centralized management of public processes. These approaches are suitable to manage business processes in B2B relationships based on e-marketplaces, where an architecture hub-and-spoke is used. The last approach, *Public and Private,* separates the definition of public processes from the private processes and carries out a decentralized management of public processes. However, it uses a proprietary low-level communication protocol to synchronize instances among WfMSs.

Web Service Composition is about orchestration of Web Services. Enterprises offer Web Services that applications running in other enterprise could invoke. Enterprises expose its functionalities (activities) through Web Services and agree on a public process that defines the control flow of these Web Services. Examples of platforms for composing Web Services are eFLOW and SELF-SERV. In eFLOW [4] the process execution is based on a centralized process engine, which is not suitable to several collaborative models. Instead, SELF-SERV [2] uses peer-to-peer interactions among the nodes that expose Web Services for the execution of the processes. However, Web Services Composition has similarities with bottom-up workflow approaches.

Modeling of public processes with workflow or Web Services composition approaches focuses on the activities or services performed by enterprises. This requires defining interdependencies and control flows of activities or services. Some activities represent information exchange and other represent enterprise tasks required to produce or consume the exchanged information. The last type of activities can be abstract, implying the invocation of private processes. However, it is not suitable to define these activities in public processes because they are private aspects of each enterprise. Enterprises could define in different ways the necessary activities with their interdependences and control flows. Activities only need to be defined are those representing the information exchange.

In contrast, when we model a public business process with an interaction protocol, we focus on the messages exchanged by enterprises and the messages orchestration. In this way, activities or services that each partner has to perform to send or receive messages are not defined in interaction protocols and are kept hidden to its other partners. Therefore, interaction protocols enable greater enterprise autonomy because each enterprise hides its internal activities, services and decisions required to support public processes.

Besides, B2B interactions do not require representing the five perspectives supported by a workflow model: functional, behavioral, informational, operational and organizational [10]. All these perspectives are important to support private processes but not to support public processes. Operational perspective is not required in public processes. The knowledge of private applications an enterprise uses to support public processes is not suitable. The knowledge and visibility of these private resources by the enterprise partners reduces the enterprise autonomy. The functional perspective, which specifies the activities of the workflow, is not also required according to the above discussed. Instead, an interaction protocol includes only the perspectives required in B2B interactions. A behavioral perspective is supported through the definition of a message control flow and an internal control flow of the roles. The informational perspective is supported through the definition of the message content. The organizational perspective is also included since the roles performed by enterprises in B2B relationships are specified.

From the information systems point of view, interaction protocols enable enterprises to interoperate while they keeping independent Interaction Protocol Management Systems (IPMSs) [18]. Enterprises do not need to use the same IPMS type. The independence of IPMSs is achieved exchanging messages through a B2B standard and by using the same language to define interaction protocols, which enable IPMSs to interpret and execute the same interaction protocols. In addition, IPMSs do not require transferring instances because messages, which have to be sent or received in each execution step, are known according to the protocol definition. Instead, workflow approaches requires that enterprises use the same WfMS because different WfMSs cannot interoperate since they use a different semantics to define the control flow [11]. Therefore, workflow instances cannot be transferred among WfMS. A proposal to solve this problem is to use a communication protocol to synchronize instances among WfMSs [5]. However, this communication protocol is a proprietary low-level protocol that WfMSs of the enterprises have to implement.

5. Conclusions and Future Works

Interaction Protocols provide a suitable approach to model and manage public business processes that are agreed by autonomous partners to achieve a goal in B2B relationships. They provide the high abstraction level required to model interactions representing public business processes. This approach also enables the implementation of information systems that fulfill the requirements imposed by decentralized collaborative e-business models.

In this work, we have discussed the design of public business processes with interaction protocols. We have discussed different aspects required to represent interaction protocols in B2B relationships. An important aspect is the use of an appropriate semantic for messages. We have selected the FIPA communicative acts library to define interaction protocol messages. In this way, we have a rich set of message types, which have an understood semantic and are based on formalisms. The use of communicative acts allows defining interaction protocols without taking into account the B2B standards that will be used to implement them. In this way, we can achieve independence between interactions protocols and B2B standards. This independence enables the reuse of interaction protocols, because enterprises can implement the same interaction protocol with different partners using different B2B standards. In addition, this independence provides a more abstraction level, because we can concentrate on the intentions expressed by enterprises to negotiate in public processes, without to taking into account details of implementation. The use of communicative acts also enables multi-agent systems technology and B2B standards to be brought together, for providing advanced solutions to B2B environments. This also allows enterprise systems that support B2B relationships to understand the same interaction protocol independent of the different technologies used to build them.

When we model interaction protocols, the use of graphical modeling languages enable interaction protocols can be modeled and understood in an easy and intuitive way. We have described the use of AUML as graphical language. The most important feature of AUML is its expressiveness, which allows a better understanding of the public processes modeled with interaction protocols. Moreover, with AUML we can specify protocol templates, which allow the reuse of interaction protocols.

Beside, from three different collaborative e-business models, we have described three examples of public processes modeled with interaction protocols. As we can observe in these examples with respect to the original definition based on workflow, modeling with interaction protocols we focus on the messages that enterprises exchange. We do not need focus on the activities or services that enterprise define to support the public processes. The hiding of these private aspects of the enterprises to its partners enables greater enterprise autonomy. This allows enterprises to define in an independent way the activities and their control flows required to send or receive messages.

Although we have discussed several aspects to model interaction protocols in B2B relationships, there are others that are required to study. One of them is the use of a suitable machine-processable language to define interaction protocols. Another issue is the incorporation of verification and validation techniques for modeling interaction protocols. These issues are part of future works.

References

[1] Bauer, B., Müller, J.P. and Odel, J. Agent UML: A Formalism for Specifying Multiagent Software Systems. *Int. J. of Software Engineering and Knowledge Engineering*, Vol. 11, No. 3, pp.1-24, 2001.

[2] Benatallah, B., Dumas, M., Sheng, Q.Z. The SELF-SERV Environment for Web Services Composition. *IEEE Internet Computing*, IEEE Society, Jan/Feb issue, pp 40-48, 2003.

[3] Bussler, C. The Role of B2B Engines in B2B Integration Architectures. *ACM SIGMOD Record, Special Issue on Data Management Issues in E-Commerce*, vol. 31, no. 1, March 2002.

[4] Casati, F., Ilnicki, S., Jin, L.J., Krishnamoorthy, V. and Shan, M.C. Adaptive and dynamic service composition in eFlow. *Proc. of Int. Conference on Advanced Information Systems Engineering (CAiSE)*, Springer Verlag, Stockholm, Sweden, 2000.

[5] Chen, Q. and Hsu, M. Inter-Enterprise Collaborative Business Process Management. Proceedings of International Conference on Data Engineering (ICDE 2001), Germany, 2001.

[6] EIDX. Replenishment Scenario 4 - Forecast-based Supplier-Managed Inventory (SMI), Version 1.0. http://www.eidx.org/publications/business_models/replenishment/replmodl4.html, Current 2003.

[7] Ericksson H-E. and Penker, M. Business Modeling with UML. John Wiley & Sons, 2000.

[8] FIPA (Foundation for Intelligent Physical Agents). FIPA Communicative Act Library Specification. http://www.fipa.org/specs/fipa00037/, 2002.

[9] FIPA (Foundation for Intelligent Physical Agents). FIPA Interaction Protocol Specifications. http://www.fipa.org/repository/ips.php3, 2002.

[10] Jablonski, S. and Bussler, C. Workflow Management: Modeling Concepts, Architecture and Implementation. *International Thompson Computer Press*, London, UK, 1996.

[11] Kiepuszewski, B., ter Hofstede, A.H.M., and van der Aalst, W.M.P. Fundamentals of Control Flow in Workflows. Acta Informatica, to appear, 2003.

[12] Medjahed, B., Benatallah, B., Bouguettaya, A., Ngu, A.H.H. and Ahmed, K.E. Business-to-Business Interactions: Issues and Enabling Technologies. *The VLDB Journal*, Springer, to appear, 2003.

[13] OMG. Unified Modeling Language (UML) Specification, Version 1.4. September 2001.

[14] Paurobally, S., Cunningham, J. and Jennings, N.R. Developing Agent Interaction Protocols Using Graphical and Logical Methodologies. Workshop on Programming Multi-Agent Systems: languages, frameworks, techniques and tool (ProMAS 2003). AAMAS 2003. Australia, 2003.

[15] RosettaNet. Collaborative Forecasting Process Scenarios. *RosettaNet Implementation Guide*, 2002.

[16] Villarreal, P., Caliusco, M.L., Galli, M.R., Salomone, H. and Chiotti, O. Descentralized Process Management for Inter-Enterprise Collaboration. *Vision, Special Issue on Supply Chain Management, Vol 7, pp. 69-79. Editor: B.S. Sahay, Management Development Institute, Gurgaon, India* (2003).

[17] Villarreal, P., Caliusco, M.L., Zucchini, D., Arredondo, F., Zanel, C., Galli, M.R., and Chiotti, O. Integrated Production Planning and Control in a Collaborative Partner-to-Partner Relationship. In S. Sharma & J. Gupta (eds.), *Managing E-Business in the 21st Century*. Heidelberg, Australia. (in press)

[18] Villarreal, P., Salomone, H.E. and Chiotti, O. Managing Public Business Process in B2B Relationships using B2B Interaction Protocols. XXIX Conferencia Latinoamericana de Informática (CLEI 2003), Bolivia, 2003.

[19] Workflow Management Coalition (WfMC). www.wfmc.org